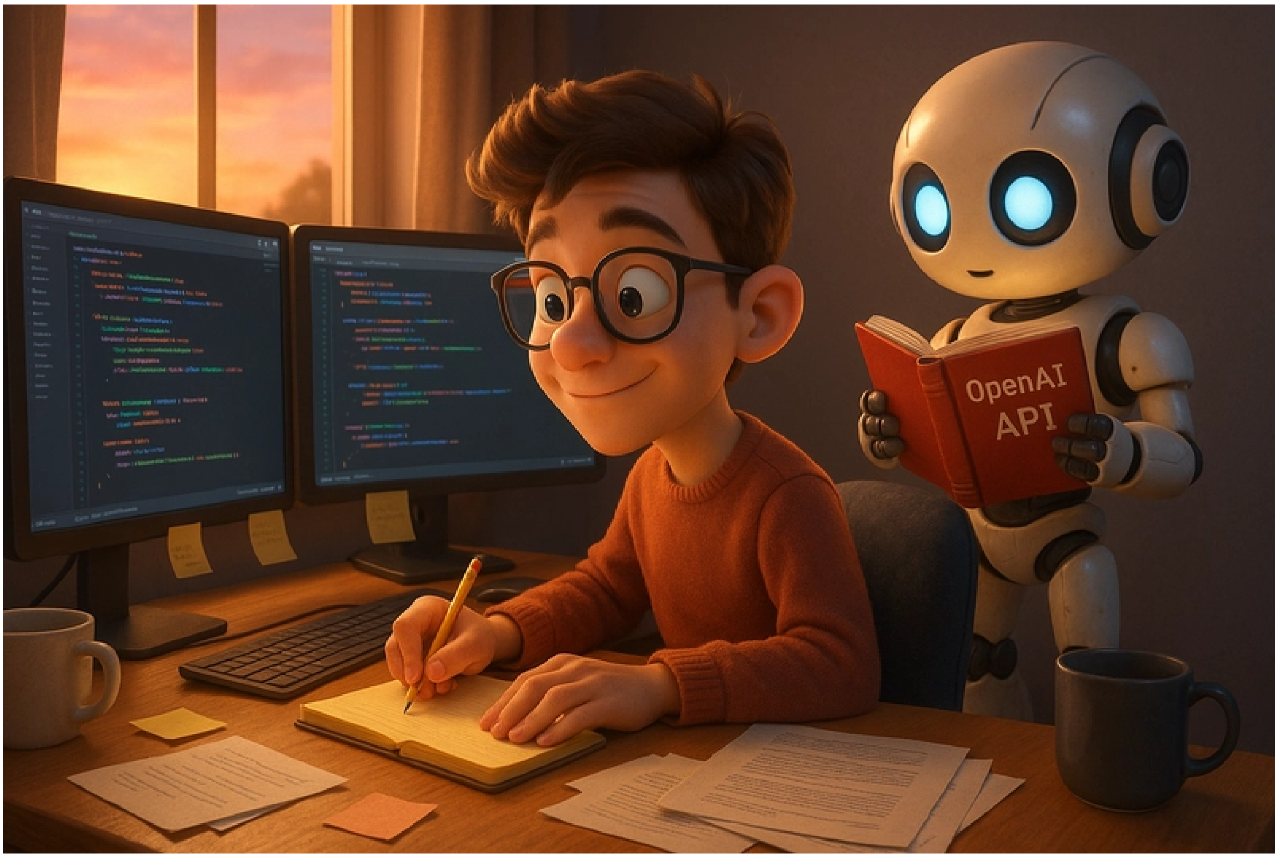

> ✨ *Want to create your own chatbot like ChatGPT? It’s easier than you think!*
>
> In this guide, we’ll walk through building a **simple yet functional ChatGPT clone** using the OpenAI API and basic web technologies.

Whether you’re experimenting or building a real assistant, this project will give you hands-on experience with OpenAI's capabilities.

🛠️ Tech Stack

- HTML + CSS + JavaScript (Frontend)
- Node.js with Express (Backend)
- OpenAI API (GPT-3.5 / GPT-4)

🔑 Step 1: Get Your OpenAI API Key

1. Visit https://platform.openai.com/
2. Sign up / log in.
3. Generate an API key from your dashboard.

> ⚠️ Keep this key secret! Never expose it on the frontend.

📁 Step 2: Setup Backend (Express Server)

Create a simple Node.js project:

```
mkdir chatgpt-clone
cd chatgpt-clone
npm init -y
npm install express cors openai dotenv
```

Create a `.env` file:

```
OPENAI_API_KEY=your_key_here
```

Create `server.js`:

```
const express = require('express');
const cors = require('cors');
const { Configuration, OpenAIApi } = require('openai');
require('dotenv').config();

const app = express();
app.use(cors());
app.use(express.json());

const configuration = new Configuration({
  apiKey: process.env.OPENAI_API_KEY,
});
const openai = new OpenAIApi(configuration);

app.post('/chat', async (req, res) => {
  const { message } = req.body;

  const completion = await openai.createChatCompletion({
    model: 'gpt-3.5-turbo',
    messages: [{ role: 'user', content: message }],
  });

  res.json({ reply: completion.data.choices[0].message.content });
});

app.listen(3000, () => console.log('Server is running on port 3000'));
```

🌐 Step 3: Create the Frontend (HTML + JS)

Create `index.html`:

```
<!DOCTYPE html>
<html>
<head>
  <title>ChatGPT Clone</title>
  <style>
    body { font-family: Arial; padding: 20px; }
    #chat { margin-top: 20px; }
    .message { margin: 10px 0; }
  </style>
</head>
<body>
  <h1>ChatGPT Clone</h1>
  <textarea id="userInput" rows="3" cols="50" placeholder="Ask something...">
</textarea><br>
  <button onclick="sendMessage()">Send</button>
  <div id="chat"></div>

  <script>
    async function sendMessage() {
      const userInput = document.getElementById('userInput').value;
      const chatDiv = document.getElementById('chat');

      const response = await fetch('http://localhost:3000/chat', {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify({ message: userInput })
      });

      const data = await response.json();
      chatDiv.innerHTML += `<div class="message"><b>You:</b> ${userInput}</div>`;
      chatDiv.innerHTML += `<div class="message"><b>Bot:</b> ${data.reply}</div>`;
    }
  </script>
</body>
</html>
```

🚀 Run Your Project

1. Start the backend:

```
node server.js
```

1. Open the `index.html` file in a browser.
2. Enjoy chatting with your own ChatGPT-style bot!

✔️ Bonus Tips

- Use `gpt-4` instead of `gpt-3.5-turbo` if you have access.
- Add conversation history to make replies contextual.